**Реферат**

Отчет 120 с., 61 рис., 1 табл., 21 ист., 3 прил.

Уважаемые председатель, уважаемые члены Государственной аттестационной комиссии, вашему вниманию предоставляется дипломная работа на тему «Разработка информационной системы интернет магазина». Актуальность темы обусловлена необходимостью разработки программного продукта, который позволяет миллионам людей, не выходя из дома, совершать покупки в интернет-магазинах. Так же данная тема актуальна для организаций, осуществляющих услуги в сфере дополнительного образования. 2020 год очень негативно сказывается на субъектах предпринимательской и государственной деятельности из – за COVID-19 тем самым повышая актуальность данной темы.

Объектом научного исследования в данной работе является интернет- магазин курсов:

В рамках работы были сформулированы следующие цели и задачи:

1. Провести анализ средств разработки и выбрать инструментальную систему для разработки интернет-магазина.
2. Разработать документно-оборотную, не реляционную, базу данных MongoDB с помощью инструментов: платформы NodeJS, Framework Express и OMD(Object Document Mapper – объектно-документный отобразитель ).
3. Реализовать классы и модули информационной системы и представить их в виде структурной системы UML, продемонстрировать при этом все сущности предметной области, атрибуты функции и связи.
4. Реализовать вэб – интерфейс интернет-магазина.
5. Тестирование приложения

**В главе 1 «Анализ объекта»** произведено описание предметной области, построена концептуальная модель информационной системы интернет-магазина.

**В главе 2 «Постановка задачи»** определены требования к информационной системе интернет-магазина. Рассмотрены пользовательские сценарии на основе use-case диаграмм, а также, составлены спецификации требований в виде пунктов и подпунктов. При анализе данных средств сделан вывод что использование use-case диаграмм является не самым лучшим способом.

Рассмотрим на примере Пример UC-6 Страница «Курсы» авторизованного пользователя, и спецификации требований в виде пунктов.

При составлении спецификаций требований был сделан вывод что данный вариант лучше т.к. некоторые детали в блок схеме рисовать крайне сложно, а также по этим спецификациям можно проводить тестирование системы. Если на каком – то этапе возникают сложности, тестировщик может указать в каком пункте или подпункте спецификаций требований баг.

**проведен обзор и обоснование выбора инструментальных средств.**

**В глава 3 «Проектирование»** рассмотрена архитектура программного продукта на основе паттернов MVC и HMVC. Как и многие парадигмы или шаблоны в программировании, MVC – это скорее общая концепция, нежели специфическая техника. Подход, который я использовал, в основном таков: я лишь делал его более формальным, назвав обработчики маршрута контроллером и отделить маршрутизацию от функциональности. Так же я представил концепцию модели представления, которую я считаю весьма важной для сохранения целостности модели. Так как корневая структура файлов и папок получилась объемной не предоставляется возможным отобразить данный паттерн на слайде.

При проектировании структур хранения данных была рассмотрена не реляционная база данных MongoDB. Для проектирования БД я использую специальный инструмент Mongoose (Object-Document Modeler (ODM)) Один из ключевых принципов – возможность управления моделью данных изнутри приложения. Не нужно возится непосредственно с базами данных, внешними фреймворками или реляционными подпрограммами отображения – можно просто описать модель данных, находясь в комфортных условиях своего приложения.

1. В MongoDB каждая запись в БД называется документом.
2. В MongoDB набор документов называется коллекцией.
3. В Mongoose описание документа называется схемой.
4. Каждая отдельная сущность данных в схеме называется путем.

Пример рис 3.7

Структурной единицей MongoDB является документ формата BSON(Binary JavaScript Object Notation - двоичная нотация объектов JavaScript) т.е. это формат электронного обмена цифровыми данными cтруктурной единицей которого в MongoDB является хранилище ключей и значений. В схеме Mongoose путем к документу является уникальный \_id, первичный ключ для каждого документа. Генерируется автоматически. Базовая структура описания пути представляет собой название пути со следующим за ним объектом свойств т.е. ключ и его значение. Во время работы с данными приложение не взаимодействует непосредственно со схемой – взаимодействие происходит через модели. Модель в Mongoose – это скомпилированная версия схемы. Однократно скомпилированный, отдельный экземпляр модели отображается непосредственно на отдельный документ в БД. Именно благодаря этому взаимному однозначному соответствию модель может создавать, читать, сохранять и удалять данные.

Пример рис 3.10

**В главе 4 «Реализация»**, рассмотрена реализация программного средства с помощью классов и модулей информационной системы, которые представлены в UML диаграммах в которых продемонстрированы все сущности предметной области, атрибуты функции и связи.

При разработке интерфейса программного продукта было рассмотрено повышение юзабилити приложения. В юзабилити как нигде кстати подходит выражение “Встречают по одежке а провожают по уму”. Заключается это в том, что любого пользователя, посещающего сайт, в первую очередь привлекает дизайн. Вторым аспектом привлекательности сайта является простота его в использовании. За это, как раз таки, отвечает качественно продуманная логика приложения в которую входят сообщения об ошибках, валидация формы и т.д.

**В главе 5 «Интеграционное тестирование»** рассмотрено тестирование приложения, которое я проводил в соответствии с составленными ранее спецификациями требований. Для тестирования я использовал фреймворк Selenium WebDriver. Результаты тестов выявленных ошибок представлены в таблице.

**В главе «Экономическое обоснование эффективности внедрения»** обосновывается эффективность использования интернет магазина.

1. Из – за того что интернет-магазин можно посетить из любой точки земного шара, существенно расширяется рынок сбыта.
2. Нет необходимости платить за аренду помещения, коммунальные платежи, оплачивать налог на имущество, страхование.
3. Ассортимент магазина можно обновлять мгновенно настолько часто на сколько необходимо.
4. Нет необходимости размещать рекламу офлайн, тратиться на изготовление и распространение печатных материалов.
5. Интернет-магазин не имеет ограничение по месту. По этому его ассортимент может приближаться к бесконечности.
6. Покупатель может совершать покупки в любой стране, 24 часа в сутки 7 дней в неделю.

Интернет – магазины медленно, но уверенно захватывают рынок становясь монополистами в товарно-денежных отношениях.

**В разделах «Промышленная экология» и «Охрана труда»** были рассмотрены опасные и вредные факторы, влияющие на окружающую среду, рассмотрены меры для обеспечения безопасной работы, организация работы по охране труда, требования к организации рабочего места при работе на персональном компьютере, электробезопасность при работе за компьютером, а также противопожарная безопасность.

**В главе 9 «Ресурсосбережение»** рассматриваются различные настройки схем электропитания, а также режимы «Сон» и «Гибернация» компьютера.

Подводя итог, можно отметить, что «Информационная система интернет магазина» имеет фактическое применение в сфере продаж.

Таким образом, поставленные задачи для разработки дипломного проекта считаю полностью выполненными.

Спасибо за внимание доклад окончен!!!

Защита приложения.

После того как я добавляю метод авторизации пользователя т.е. метод регистрации и метод логина с приложением может взаимодействовать лишь один пользователь. Для того что бы с приложением могло взаимодействовать множество пользователей я добавляю сессии.

Для того что бы пользователь не мог редактировать курсы другого пользователя я каждому из пользователей предоставляю определенные права доступа. Если пользователь не является владельцем курса, то кнопка редактировать, от данного пользователя скрыта. Но просто скрыть кнопку мало так как если вбить правильный запрос в адресную строку можно перейти на страницу редактирования. Для этого нужно защитить редактирование на стороне back-end. Для того что бы это не происходило я сверяю id пользователя, который находится в объекте курса, с id пользователя находящегося в сессии.

Далее для того, чтобы защитить роуты программно я использую middleware auth.js и результат данной функции добавляю почти в каждый запрос. При запросе происходит проверка авторизован ли пользователь в системе или нет. Если же нет происходит перенаправление на страницу логина.

Если злоумышленник находит лазейку он может тем или иным способом выгрузить все данные из БД, тем самым он получит логины и пароли всех пользователей. Однако если пароли в БД хранятся в зашифрованном виде это станет серьезным препятствием для злоумышленника.

Для шифрования пароля я использую пакет bcryptjs.

Хоть сессии и хранятся в БД однако злоумышленники могут использовать к примеру ссылки из email которые позволяют перехватывать сессии. Для защиты приложения я использую пакет csrf с помощью которого происходит генерация токена на стороне сервера. Данный токен помещается в cookie, например X-CSRF-TOKEN, и помещается в объект header.

При отправке пользователем запроса на сервер, данный токен из cookie попадает в форму и в месте с запросом отправляется на сервер, где происходит проверка на идентичность токена из данных сессии и токена который прислал пользователь.

Если оба токена совпадают, то запрос не подтвергается CSRF-атаке, в противном случае логируется событие и отклоняется запрос.

Если запрос не отклонился, то на сервере генерируется новый токен который будет использован в последующие запросе.